THE AUTOMATED ASSEMBLY OF SIMULATION MODELS

(This paper is UNCLASSIFIED)

Colonel Thomas B. Roelofs, USA (Ret)
Operations Analysis Division, General Research Corporation
Westgate Research Park, McLean, Va. 22101 (703-893-5900)

NTRODUCTION

Most operations research analysts have, at one time or another, anted to use a simulation model for the analysis of a problem being tudied. Research of available models usually reveals that the formuation needed hasn't been developed, or an existing model doesn't match he problem. Faced with this situation he could modify the problem to atch the available model, usually an unwise alternative, modify the vailable model, or develop a model to match the problem. The formulaion and implementation of complex simulation models until recently has equired a lot of talent, a lot of effort and a lot of time. Modifying xisting models can be as costly as developing one. Many studies that ould have benefited from the application of simulation models by-passed hat route because of lack of time or resources.

This paper will describe a system that has recently been developed by the General Research Corporation which provides the capability to automate the assembly of simulation models. The system was developed for the US Army Logistics Doctrine, Systems and Readiness Agency, a Class II Activity of the Department of the Army Deputy Chief of Staff for Logistics. It has been tested and proven to operate satisfactorily.

BACKGROUND

Several years ago the Research Analysis Corporation (RAC—the predecessor to the Operations Analysis Division, General Research Corporation) was awarded a contract by the US Army which included the task of developing a model of the US Army worldwide logistic system. The development study was given the acronym "MAWLOGS" (Model of the US Army Worldwide Logistic System). The task specified that the formulation was to be a simulation model to be used to "compare proposed systems with each other and with the current system to determine the relative merits of each system."

The research that preceded developing an approach to the modeling task revealed that the model had to be flexible in terms of functional range, system scope, drive, and level of detail and be structured along the lines of a node network system, which is characteristic of logistic systems. The logistic problems that would need to be addressed by such a model ranged from studying a single function within a small segment of the system to those of a worldwide nature, usually multifunctional in scope, and multi-item in detail. Thus, the model would have to be driven by demands for various types of logistic support, such as the supply of materiel and the provision of maintenance, originating at any echelon in the system, ranging from the troop unit level to the national level. Most of the problems identified during the course of the research focused

on less than the total system, involved two or more interacting functions (e.g., supply, maintenance and transportation), and usually varied with respect to the desired level of detail. In addition, there appeared to be the need to treat one function, say supply, at one level of detail while treating the other interacting functions, such as maintenance and transportation, at different levels. A single model of the Army world-wide logistic system that would include these options appeared inefficient and indeed infeasible, considering available computers. Consequently, a flexible modeling system seemed a reasonable approach, and the task became one of designing a system for the rapid assembly of simulation models of specified scope and level of detail, designed to focus on the particular problem to be studied. Such a system has been developed and is currently operational at the General Research Corporation (GRC) facility in McLean, Virginia. In the near future, the Army is expected to develop the capability to apply the system at the Logistics Center at Fort Lee, Virginia.

The total system developed for the Army by GRC, called the MAWLOGS System, includes two computerized components in addition to those required for the rapid assembly of simulation models. These include a small set of programs designed to preprocess "raw" data for input to a model, called the Automated Input Data System, and a set of programs that process data output by a model, called the Output Data Postprocessor System. They basically support the application of a model and neither will be described in this paper. The complete documentation of the MAWLOGS System is expected to be disseminated by the Army near the end of 1973.

GENERAL DESCRIPTION

AUTASIM is the acronym for the Automated Assembly of Simulation Models. It is that part of the MAWLOGS System that creates simulation models. It represents a significant innovation in modeling methodology that should be of considerable interest to Army operations research analysts. The description of the AUTASIM system is the purpose of this paper.

The AUTASIM system consists of three elements: a Module Library, a Model Description Language and a Model Assembler program. The Model Assembler and the Model Description Language represent the primary innovations in creating simulation models. Development of a Module Library was not a trivial task, because it is broad in scope and actually required more time and effort to develop than the other two elements. Currently, it includes modular computer routines, each simulating a specific activity, and service routines that provide the conventional elements of a simulation model. Its form was dictated largely by the other two elements. Certain of the modules in the library—called verbs—constitute the vocabulary of the model description language. The Model Description Language is a methodology for describing the node network structure of a model and the activities that are to be simulated at the nodes and over the links in the model. The Model Assembler is a computer program which, given the description of the system to be modeled, will retrieve from the Module Library the required modules, link them together as prescribed in the model description, and produce a computer program of the system model.

SYSTEM CHARACTERISTICS

The AUTASIM system assembles simulation models of systems that can be described as node networks. Nodes are centers of activity and links are communication and transportation paths between nodes. The models can vary greatly in size and the scope of activities simulated. A model may contain one or more system nodes. The activities that can be simulated are represented by the modules in the Module Library. If an activity is needed in a model that is not represented in the Module Library, it is relatively simple to develop the required module or modules and add them to the library.

The activities simulated at each node in the system modeled are defined as an activity network of nodes and links, when more than one activity is simulated at a node. Each activity node can also be defined as a subactivity network. Figure 1 shows how a system can be represented as different levels of module networks. The content of other nodes may include the same modules but the network can be different. Modularity in this form provides great flexibility in defining the activities to be simulated at the desired level of detail. It also facilitates inclusion in a model of only those features that will accomplish the purpose of the model, with no extraneous logic and waste of core storage.

The AUTASIM system is fully automated. The Module Library is written on a tape file. Given the description of a model to be assembled, the Model Assembler program can assemble a model in a very brief period of time—five to ten minutes. This rapid model assembly capability also facilitates rapid changes to the activity content or system structure of a model, a characteristic often needed when comparing alternative concepts. The computer programs included in the AUTASIM system, the Model Assembler and the modules in the library, are written to the maximum extent feasible in USA Standard FORTRAN.

MODEL CHARACTERISTICS

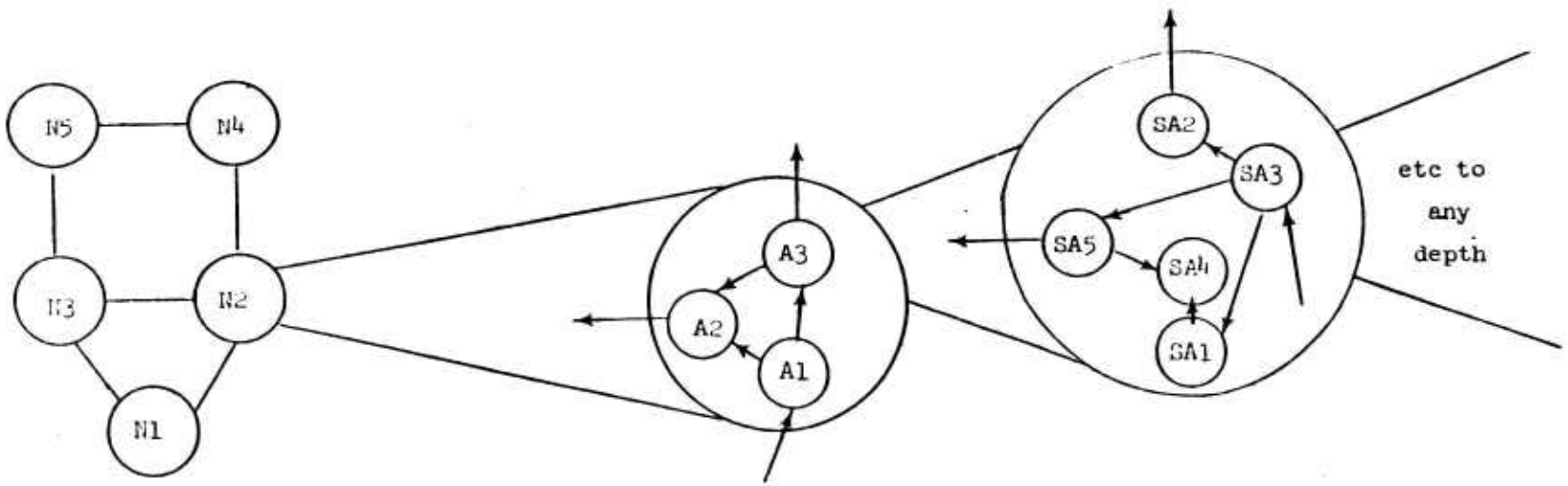
A model produced by the AUTASIM system can be described as a discrete-event, dynamic simulation model. Most AUTASIM models are also stochastic; however, all elements of uncertainty in a model may be omitted by the model designer, which would then yield a deterministic simulation model.

The model programs generated by the Model Assembler are written in USA Standard FORTRAN.

Models assembled by this system have warmup and restart capabilities. The former shortens model running time before statistics collection is begun. The latter permits saving the status of a model at any point in simulation time for future model restart at that point in time. This permits the analysis of collected statistics at frequent intervals to determine their adequacy and can conserve computer usage.

Every model includes a complete statistics collection capability. The statistics the user wishes to collect and the form in which they are to be recorded are specified in the input data deck for model execution. A generalized reporting capability is also included in each model. The

OVERVIEW OF MODULAR APPROACH



SYSTEM:	NODE:	ACTIVITY:
Node Network	Activity Network	Subactivity Network

Fig. 1

user is free to specify when the model is assembled the reports he requires from the model. Statistics may also be collected in detail on a tape file for postprocessing.

As an aid in verifying the model, a trace capability is included. Prior to full-scale execution of a model, it is prudent to ensure that the model does, in fact, represent the desired system. This is usually a tedious and time consuming job, which is simplified by the built-in capability to trace events through all or parts of the system modeled.

MODEL DESCRIPTION

A model is described by using a symbolic language that facilitates the description of very complex node networks and is accepted by the Model Assembler program. The language consists of a "vocabulary" of module names, a set of delimiters, and a set of conventions for combining module names and delimiters to define the content and structure of a model. Experience has shown that this capability by itself is a very powerful tool. Multifunctional networks so complex as to be almost impossible to draw in the form of node network diagrams, can be described in the Model Description Language.

An illustration of the use of the language in describing a model will be given following the definition of some terms that have special meaning in the context of the AUTASIM system. These terms are listed below.

System	- a network of nodes connected by links.
Node	- a special block of programming logic which can be referenced in a model. An activity center.
Verb	- any block of programming logic which can be included in a model description.
Simple verb	- a block of FORTRAN code. The logic for simulating an activity.
Nonsimple verb	- a structured assemblage of simple verbs into a larger block of logic.
Module	- any block of logic which is contained in the Module Library; the set of modules includes verbs, service routines, and common data structure decks.
Parameter Slot	- a point in the logic of a verb at which control may be transferred to logic outside the verb.

The verbs in the Module Library are the vocabulary of the model description language. The content of a model is the set of simple verbs, or blocks of programming logic, which are specified in the model description. The structure of a model is the way the content blocks are interconnected.

The general structure of a verb is shown in Figure 2. The header information includes a reference to the routines called by the verb and the input data requirements of the verb. The program of a verb, like most programs, consists of logical steps or parts, with some of the parts contained in subprograms that are called from within the program. In the case of a verb, all the parts need not be included in the program or even be referenced by name, since a general external reference can be made. Parameter slots provide the capability of external references that need not be prespecified. A parameter slot can be inserted in the verb program at the point where logic outside the main verb logic may be applied to complete the function of the verb. A verb that deals with reordering stock, for example, could determine the reorder quantity by any of several policies. When the reorder verb program is written, the specific reorder policy need not be known and a parameter slot can be inserted at the place in the program where such a policy would be implemented. The person describing the model can therefore specify the policy he wants implemented by filling the parameter slot with a reference, probably another verb, that simulates the desired policy. This modular form of programming provides great flexibility and minimizes the size of verb programs. The simple multinode system diagram shown in Figure 3 will be used to demonstrate how the model description language is used to describe a system. The diagram shows a five-node system. Each node must be given a name or number of from one to five characters. Recall that each node is an activity center where one or more activities are simulated. The block in Node 1 labeled A represents the activities to be simulated at Node 1, the blocks labeled A, C and D, those to be simulated at Node 2, etc. The precise activities represented by A at Node 1 and their connection with Node 2 are described in the Model Description Language as shown below.

```
NODE1. VERBA (1 = VERBB $
              2 = VERBC (1 = VERBB)),DELAY(P=3),*NODE2 $
```

This description of Node 1 states that it contains the activity represented by VERBA; that at the point in the execution of the logic of VERBA where Parameter Slot 1 is encountered, control is transferred to VERBB; that after the execution of the entire logic of VERBB, control is returned to VERBA; that after the execution of some more logic of VERBA where Parameter Slot 2 is encountered, control is transferred to VERBC; that at the point in the execution of VERBC where Parameter Slot 1 is encountered, control is transferred to VERBB; that after the entire logic of VERBB is executed, control is returned to VERBC; that after the remaining logic of VERBC is executed, control is returned to VERBA; that after the remaining logic of VERBA is executed, control is transferred to verb DELAY, which has been directed to find a value in probability distribution number 3 for the delay parameter in that verb; that after verb DELAY is executed, control is transferred to Node 2. Schematically this description would appear as shown in Figure 4.

The significance of this illustration is that it demonstrates how one can describe a model including very complex interrelations among blocks of logic with time interdependencies among system processes in relatively simple form. It also demonstrates the flexibility in varying logical procedures available to the person describing a system to be modeled. Admittedly he must be very familiar with the system to be

VERB STRUCTURE

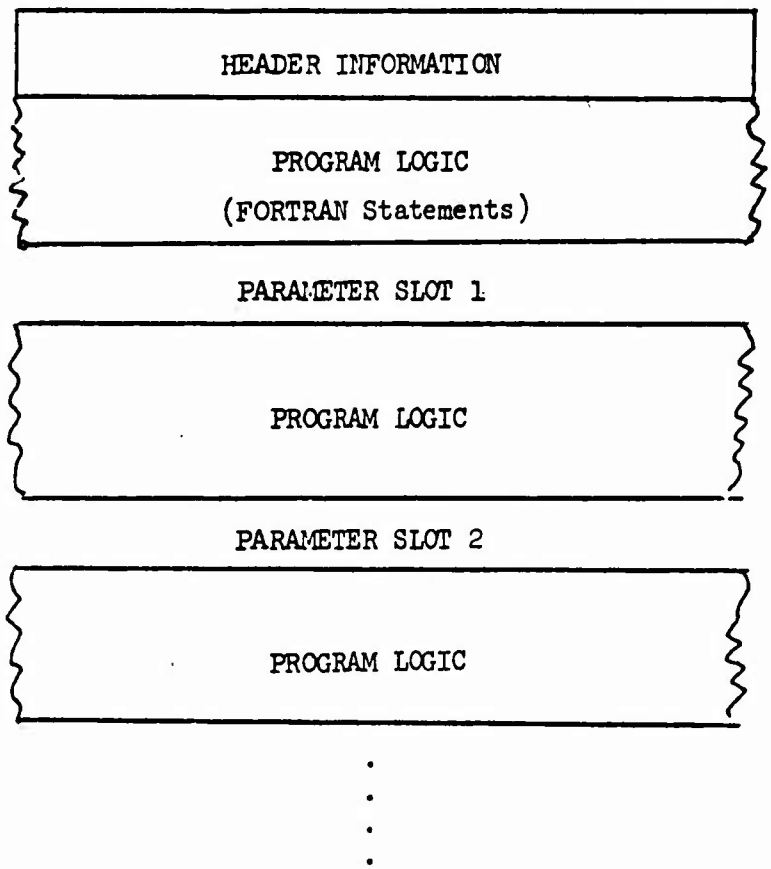


Fig. 2

ILLUSTRATIVE NODE NETWORK SYSTEM

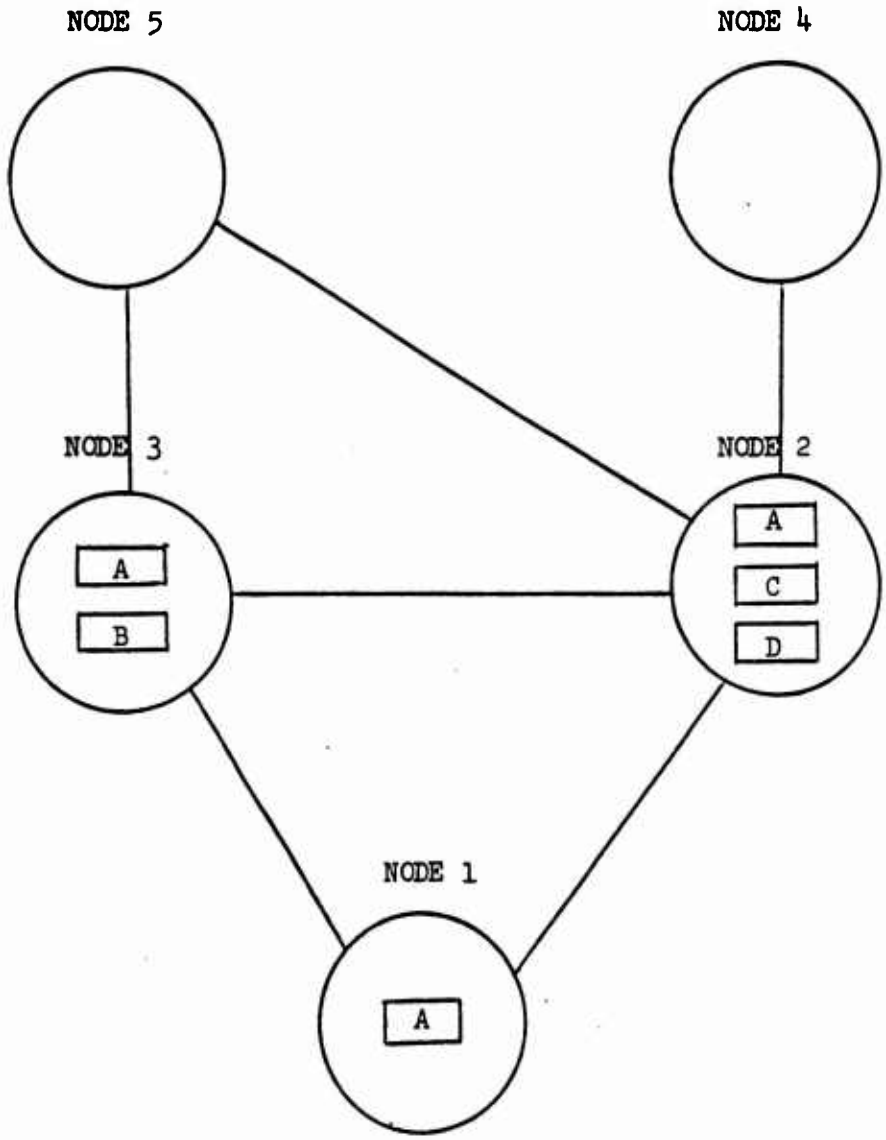


Fig. 3

SCHEMATIC REPRESENTATION OF FLOW CONTROL

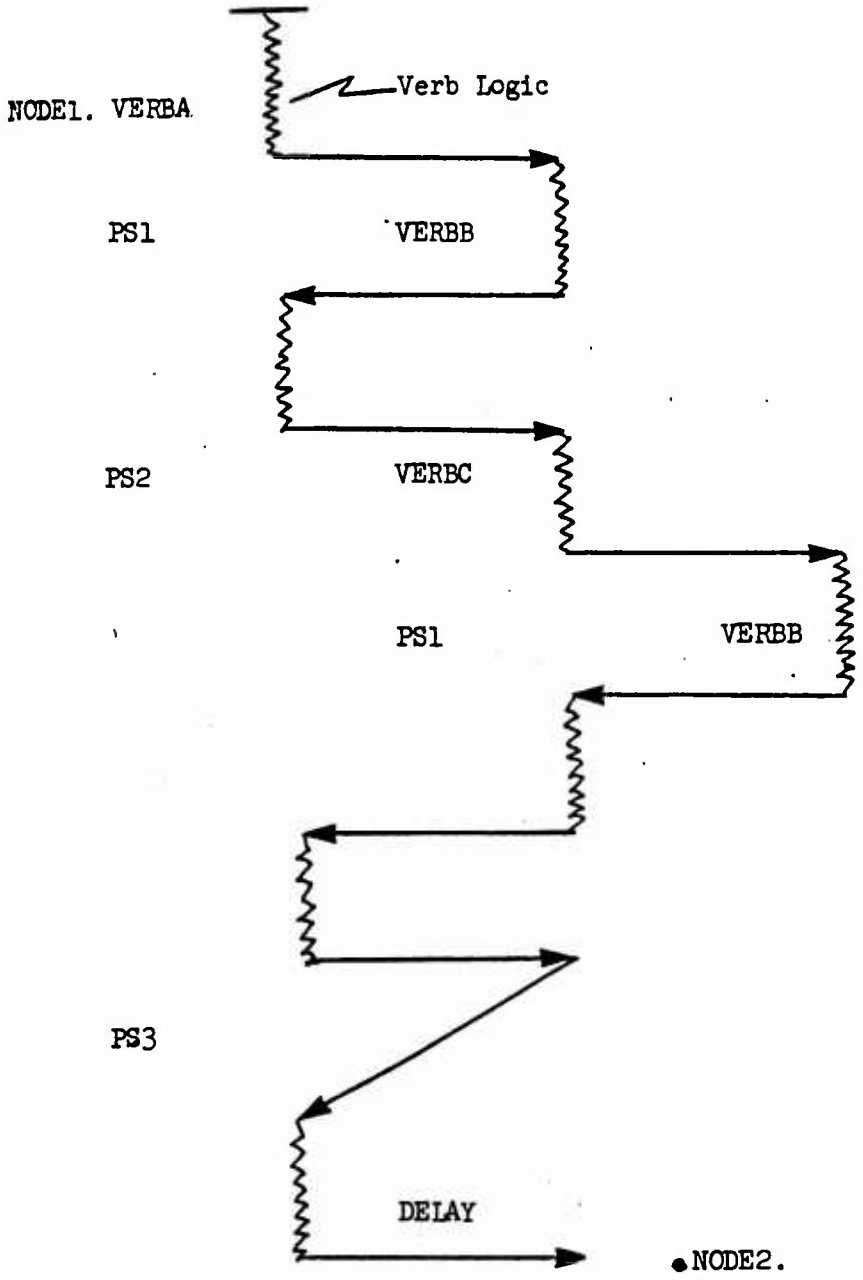


Fig. 4

modeled, which is essential for any analyst describing a model, and quite intimate with the contents of the Module Library.

Model descriptions can be simplified by the use of nonsimple verbs. Commonly used combinations of simple verbs can be formed into nonsimple verbs and be stored as such in the Module Library. The earlier example of verb VERBA in Node 1 can be constructed as nonsimple verb NSVA1 and be defined as shown below.

```
NSVA1:  VERBA (1 = VERBB $ 2 = VERBC (1 = VERBB)), ** 1
```

The key point is that NSVA1 has been defined as a particular pattern of simple verbs VERBA, VERBB and VERBC and can be used in model descriptions. The description of Node 1 shown earlier can now be written as follows:

```
NODE1.  NSVA1 (1 = DELAY(P=3), * NODE2) $
```

The reference to nonsimple verb NSVA1 in the model description would cause the Model Assembler to include its structure in the model. One or more verbs used to define a nonsimple verb may themselves be nonsimple, to any depth. Every nonsimple verb however must ultimately be expandable into only simple verbs, since, among verbs only simple verbs may contain program statements that are executed.

MODEL ASSEMBLER

The Model Assembler is a computer program that, given the description of a model written in the Model Description Language and access to the Module Library, will retrieve from the library the required modules, generate linkage routines interconnecting them according to the model description, and output a complete computer program of the model. An additional input for a model being assembled is a set of dimension values that are used to set the dimensions of the data storage arrays in the model program.

The model assembly process is shown in Figure 5. The assembler scans the model description one node at a time, building a list of the designated verbs. Any nonsimple verbs are then expanded. Modules referenced by the verbs on the list are then added to the list. At the end of a node scan, linkage routines are created which connect the verbs in the specified manner. This process is repeated for each node in the model description.

After all nodes are scanned, the modules required in the model are retrieved from the Module Library and the module and linkage source code are combined. Then a list of all common data structure decks required by these modules is made, the decks are retrieved from the Module Library, and their dimension values are set.

The Model Assembler outputs include the complete model program written on a tape file, a listing of the model program, an expanded model description, a list of the modules in the model, and a list of the model input data requirements. The expanded model description includes the

MODEL ASSEMBLY PROCESS

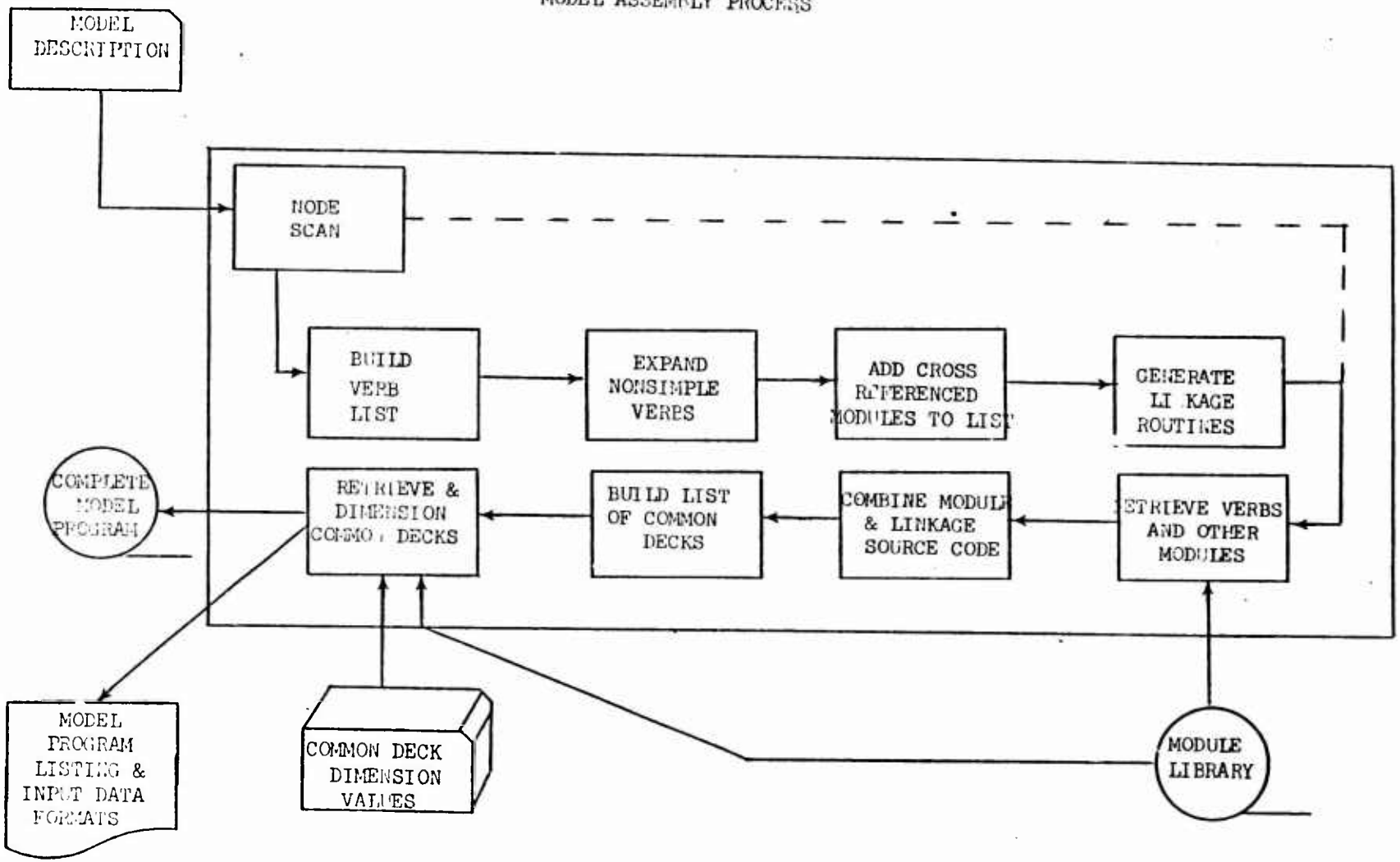


Fig. 5

description as input plus the expansion of all nonsimple verbs to show their structure. The list of input data requirements is an important and valuable output of the Model Assembler, because the inputs for a model depend on the verbs included in the model. Each model therefore requires a unique set of inputs. This listing is a great help to the model user in preparing the inputs for model execution.

Figure 6 shows how the model input requirements are described. The inputs are listed by node and within node by module in the order in which the cards are to be assembled for input. The input requirements for a module include a description of each data element, the card columns in which the data are punched, and the format for the data.

The Model Assembler program is written in FORTRAN, occupies about 32,000 words of core storage on the Control Data 6400 computer and requires nine files. Model generation times have ranged from five to ten minutes of central processor time on a Control Data 6400 computer. Small models of six to eight nodes could require the lower bound of the time, while large models of about twenty nodes could require the upper bound of time.

It should be noted that since the Model Assembler scans one node at a time, there is no practical limit on the size of the models that can be generated. Thus the assembler could produce models that would exceed the core capacity of generally available computers.

MODULE LIBRARY

A brief overview of the Module Library is needed to complete the description of the AUTASIM system. The general structure of the library is shown in Figure 7. As stated earlier it consists of verbs, service routines and common data structure decks. The verbs currently in the library are those required to assemble models of logistic systems. This part of the library is expected to grow as new verbs are developed and added to simulate activities foreign to those simulated by the current verb library. The service routines represent a complete package required for almost any model that could be assembled by this system. Common data structure decks might be expanded as new verbs are added. A point to be made is that the addition of new verbs required by models that can not be assembled from existing verbs is a trivial task compared to building a complete model in the conventional manner.

The figure does not represent the true proportion of verbs, service routines and common decks. There are about 400 modules currently in the library. Approximately 220 are verbs, 160 are service routines and 20 are common data decks.

It was decided by the Army that the initial set of verbs to be developed would be those necessary to simulate the Army material support system with supporting transportation and communications. The materiel support system can be defined as that part of Army logistics by which fleets of end items are supplied to and maintained in using units. The content of the verb section of the module library reflects that decision. Figure 8 shows the four families of verbs currently in the library.

MODEL INPUT DATA REQUIREMENTS
PRODUCED BY MODEL ASSEMBLER

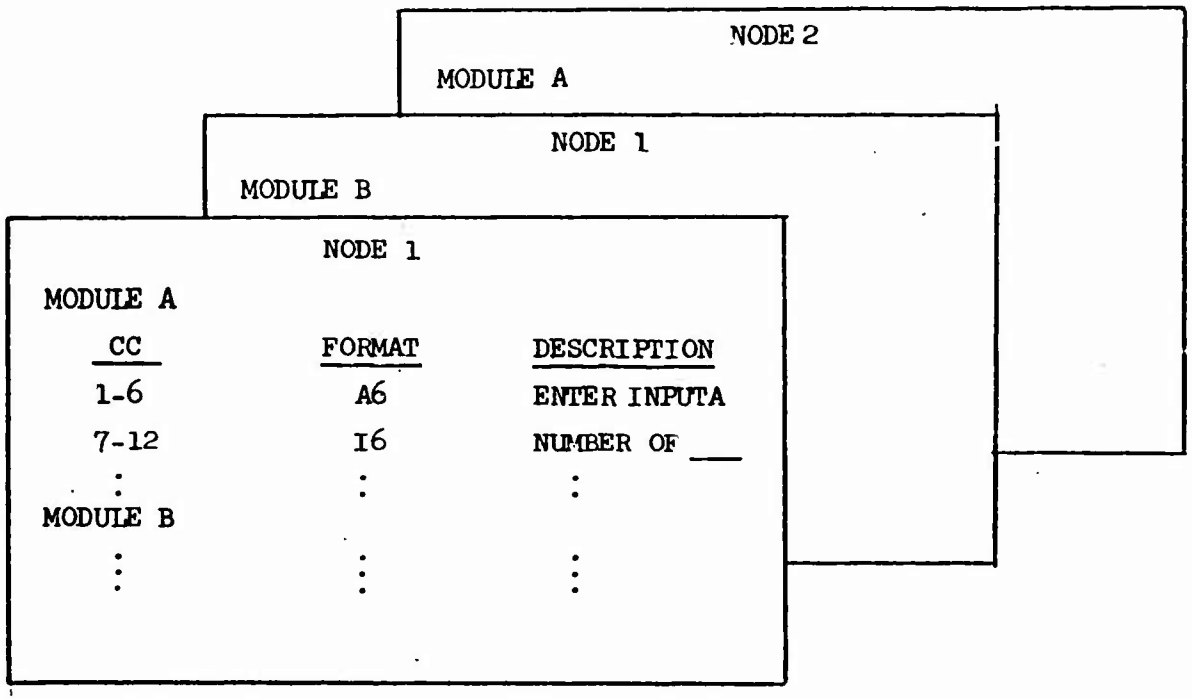


Fig. 6

MODULE LIBRARY STRUCTURE

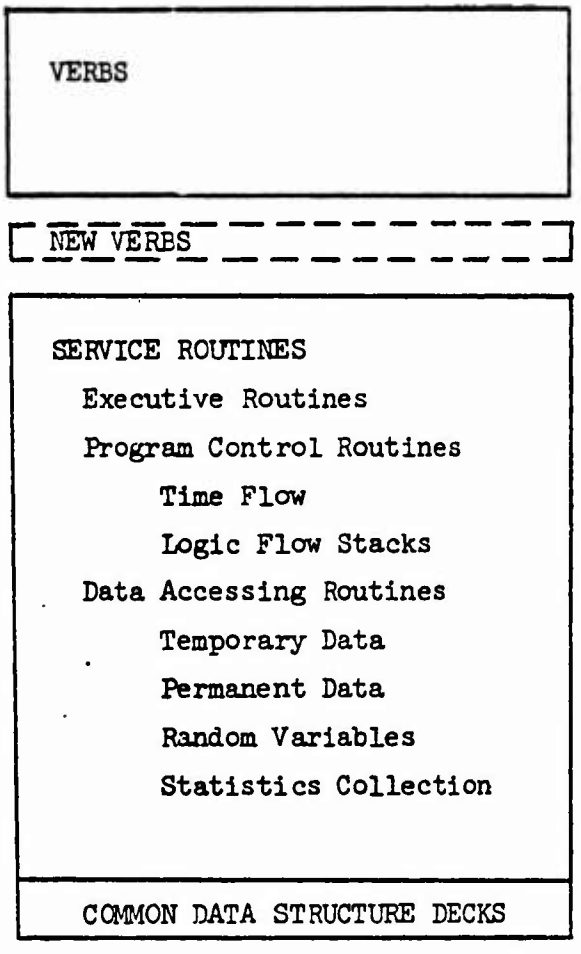


Fig. 7

EXPANDED VIEW OF VERB SECTION
OF MODULE LIBRARY

FIELD MAINTENANCE AND SUPPLY FAMILIES
- Supply
- Direct Exchange
- Maintenance
- Rebuild
- Float

WHOLESALE SUPPLY AND MAINTENANCE FAMILIES
- National Inventory Control Points
- Supply Depots
- Maintenance Facilities

TRANSPORTATION FAMILIES
- Aggregate Transportation
- Detailed Transportation

COMMUNICATION FAMILY

Fig. 8

The field maintenance and supply families simulate the material support system below the wholesale level. These verbs simulate the key supply and maintenance activities associated with the support of troop units and fleets of equipment at the troop unit level. They include the generation of maintenance demands, the repair, rebuild and salvage of end items and reparable components, and the supply of end items, components and repair parts. The level of detail is that of individual transactions, such as maintenance demands, supply requisitions and individual shipments. Statistics are collected on fleet availability, supply response, workloads, resource levels, resource utilization and time delays.

The field supply and DX activities simulated include demand processing, replenishment ordering, receipt processing, demand forecasting and revision of inventory policy parameters. The major maintenance activities simulated include diagnosis, skill assignment, parts assignment and repair. Related activities include generation of demands, ordering of parts, and issues from maintenance floats. The main difference between field maintenance and field rebuild is that in field maintenance unserviceable items are repaired one at a time, while in field rebuild they are repaired in batches.

In wholesale supply and maintenance the available levels of detail are similar to those for the field. That is, activities are simulated at the level of individual transactions, such as single requisitions, procurement orders, and supply control studies for individual items. The wholesale system is viewed as consisting of NICPs, supply depots and maintenance facilities.

Both of the transportation families simulate the movement of individual shipments over links that connect transportation terminals. Up to six different modes of transportation can be simulated in the same model. These are air, sea, rail, highway, inland waterway and transshipment from one mode to another. In the aggregate family only the movement of items with a combination of delay times for terminal operations is simulated. The detailed family simulates the consolidation of shipments into vehicle or carrier loads and the movement of the carriers through the network, assigning docks at terminals, maintaining arrival and departure queues, and diverting carriers from overloaded terminals.

The communications family is designed to simulate delays encountered in processing and forwarding individual messages through communication terminals. Various forms of message scheduling are available.

In summary, the Model Description Language has been described. It facilitates the description of very complex node network systems, multifunctional and multiechelon in nature. It is a very powerful modeling tool. The Model Assembler applies this language in assembling discrete-event dynamic simulation models. Given the required modules, it can assemble models in five to ten minutes of CP time on a Control Data 6400 computer. The combined capabilities of the Model Description Language and the Model Assembler represent significant technological achievements that should lead to greater use of simulation models in operations research studies. The content of the current Module Library provides the capability

to model a wide range of logistic systems at several levels of detail. It can easily be expanded to embrace additional functions or different levels of detail for functions currently represented. The operations research analyst looking for a model to fit his problem should consider the AUTASIM system. It is not a model in itself, but a system for the rapid creation of a wide range of simple or complex simulation models.

In closing it is appropriate that contributions to this paper by two colleagues at GRC be acknowledged. They are Dr. Robert T. Burger and Mr. Howard A. Markham. They and Mr. Thomas M. Lisi, inventor of the model assembly language and developer of the initial formulation of the model assembler program, are largely responsible for the development of AUTASIM.